Reader Spec

Technical Specification
for the
Softstrip®
Reader Interface

July 1986

Cauzin Systems, Inc.
835 South Main Street
Waterbury, CT 06706
(203) 573-0150

1 Preface

1.1 Overview

This document is intended for use by system designers wishing to use the Softstrip (R) System as part of an integrated application package. The information that follows in this document is proprietary information of Cauzin Systems, Inc., which retains all rights to this material. All specifications stated in this document are subject to change without notice.

"Data strip" is a generic term used to describe the encoding of electronic data using paper and ink. "Softstrip" is a registered trademark of Cauzin Systems, Inc., and is the brand name Cauzin uses on data strips they produce.

The structure and esthetics of Softstrip data strips are required for two purposes. One is for alignment and error correction by the optics and firmware inside the reader. The other is to facilitate transmission of data in a format that is usable by a Cauzin communications program. The format includes information for recreating files on a disk after reading from a data strip.

The developer of a communications program to interface the reader with a host computer requires the technical specifications shown in this document to utilize the Cauzin Softstrip Reader.

It is necessary to know the reader's communications protocol and the field layouts. A casual understanding of the internal electronics and mechanical mechanisms is required only to better decipher the error conditions that can occur during a data strip read.

2 Reader Interface

2.1 Overview

The Cauzin Softstrip Reader (referred to as the "reader") has an RS-232 port to communicate with a host processor. Once the reader starts a normal read, it closes its "ears" to the host processor, demanding that the host receive all its transmitted data until the reader recognizes the end of a strip. The reader cannot be "XOFFed" like a modem.

At the end of a strip or if an error is detected, the reader sends a command sequence to the host and returns to the start position. The reader's "ears" remain closed until it reaches the start position, where it looks for a command from the host computer.

2.2 Communications Specifications

- **4800 Baud**
- **1 Stop Bit**
- **No Parity**
- **8 Data Bits**
- **Break (or CTF)**

While the serial speed specifies 4800 baud, the effective data speed is slower, depending on strip density. The throughput ranges from 1000 bits/second at low densities to almost 2000 bits/second at high densities.

A CTF (Command-To-Follow) is a 74 millisecond low pulse (space), followed by a 74 millisecond high pulse (mark). A host processor may detect a CTF as a BREAK or framing error, and should continue to wait for a start bit to validate new data.

2.3 Reading Data Strips

2.3.1 Start-up sequence

When the reader is in the home position, it looks for a one-byte ASCII command. For a normal read sequence, the host should transmit an ASCII "R" ($52). Other command codes are documented below, but none puts the reader into an appropriate data strip interpreting mode except the "R" command.

When the reader successfully identifies the command, it immediately sends a CTF to the host, then echoes (retransmits) the command it received. In this case, the host receives an "R" following the CTF.

Upon completion of the start-up sequence, the reader makes no further attempt to recognize commands from the host. The reader begins its alignment process and starts sending strip data to the host. Only an **error** or **end-of-strip** interrupts the reader.

2.3.2 End of strip

The diagram below represents a sample start-up process, with data sent by the reader, and a successful end-of-strip signal.

Host	**Reader**
R ($52)--->	
	<--- CTF (Break)
	<--- R ($52)
	<--- {strip data}
	<--- CTF (Break)
	<--- A ($0A - End-of-strip command code)

2.3.3 Error detection

This next diagram shows a sample start-up process, data sent by the reader, and an error detected.

Host	Reader
R --->	
	<--- CTF (Break)
	<--- R
	<--- {strip data, if any}
	<--- CTF (Break)
	<--- 8 ($08 - Error command code)
	<--- 0 ($00 - Error code)

Note: The error code following the error command code has a value from 0 thru 9 as defined below.

Also take note that only a successful read sends the command code for an end of strip (CTF, $0A). The error command code (CTF, $08 , $xx) only occurs in the event of a fatal read error and never signifies the successful completion of a read operation.

2.4 Reader Error Codes

2.4.1 Overview

If the reader's firmware detects an error after the start of a strip read, it halts transmission of data to the host and issues a CTF, followed by the ERROR COMMAND CODE ($08), followed by an ERROR CODE ($00 to $09). The host processor must interpret the error code, issue an appropriate error message, and perform any internal operations required for the next read sequence.

If the error calls for restarting the reader, remember that the reader rewinds itself automatically at the end of a read or upon detection of an error. Rewinding could take up to 10 seconds, and the reader will have its "ears" closed to any command during that time.

The host must be capable of waiting up to 30 seconds for a read and 10 seconds for a rewind (40 seconds total). If the host issues any command and gets no response, it should continually issue the command (i.e., every 1/2 second) for a reasonable time period (more than 40 seconds) before issuing a time-out message. The host time-out message, different from the reader's firmware time-out, should suggest checking the power or cables.

2.4.2 0 - Corridor closed

A corridor closed error is caused by the reader unsuccessfully reading a data line after all error correction is attempted. It could be caused by:

- bumping or moving the reader during a read.

- too much ink (or carbon black mark) deposited on a strip, making the "bits" indecipherable. A pencil mark could cause this type of problem, although the reader has a good chance of successfully reading through most pencil marks.

- strip destruction, either a tear where the strip is not patched together properly or ink dropouts where bits are missing from the strip. The reader has error correction capability to survive some dropout.

Because rereading the strip may be successful, the appropriate error message would be:

"UNABLE TO READ - PLEASE RETRY"

2.4.3 1 - Horizontal sync not established

Horizontal synchronization must be established before the reader can proceed down the strip. An error occurs if the reader cannot determine the horizontal density that is coded at the beginning of the strip. Refer to the Horizontal Sync Field description section for more information.

This can be caused by poor printing or by misalignment. If the strip is poorly printed or destroyed in this area, chances are that a successful read will never take place. Because poor strip alignment causes the same problem, the following error message should be issued.

"STRIP ALIGNMENT - ADJUST AND RETRY"

2.4.4 2 - Strip too close to top

The **strip too close to top** error is issued when the reader's firmware thinks it has found the top of a data strip with insufficient room to perform its initial tilt alignment. Generally, the user need only realign the strip a little further (i.e. less than 1/4 inch) down the reader. This can be caused by stray black marks too close to the top of the strip, or by an improperly positioned alignment dot.

The suggested error message to issue is:

"STRIP ALIGNMENT - ADJUST AND RETRY"

2.4.5 3 - Invalid expansion bytes

The expansion bytes are used for future, specialized readers. These bytes currently must be coded as zero's on the strip. When non-zero expansion bytes are discovered by the reader's firmware, an error is issued to the host. Only specially modified readers with the required firmware can use these strips.

Since there is no reason to use expansion bytes in typical data strips, the host can only interpret this to mean that an unauthorized or unusable data strip has been read. The host should issue the following error message:

"UNABLE TO READ - RETRY" or optionally

"UNAUTHORIZED READER - STRIP NOT USABLE"

A misread of the expansion bytes can produce the same error. While the chance of a misread is slight, there is no reason to accuse the user of using an unauthorized reader. Let the user retry the strip a few times before giving up.

2.4.6 4 - T1 too small

This error message is caused by a tilted data strip. Reading is OK at the top of the strip, but as the reader scans down the strip, not enough room is left between the strip and the reader's edge. This is an alignment problem that can't be detected until it occurs.

The host should issue the message:

"STRIP ALIGNMENT - ADJUST AND RETRY"

Essentially, the read did not complete and the user needs to be advised of the potential misalignment.

2.4.7 5 - Vertical sync not established

Vertical synchronization must be established before the reader can proceed down the strip. This error is reported if the reader cannot determine the vertical scanning density that is encoded at the top of the strip. If the strip is poorly printed or destroyed in this area, chances are that a successful read will never take place. Refer to the Vertical Sync Field Description for more information about vertical density.

Use the following error message:

"UNABLE TO READ - PLEASE RETRY"

2.4.8 6 - Data sync not established

Data synchronization must be established before the reader can proceed down the strip. A problem occurs when the reader cannot successfully read a zero ($00) data byte immediately following the successful completion of vertical sync. The reader firmware gives up trying to find data sync after 256 scans, more than adequate time.

Since the reader has successfully scanned a number of data lines to reach this point, you can expect a successful read to take place on the next try. A detected misread must have occurred. It is suggested that the following error message be issued.

"UNABLE TO READ - PLEASE RETRY"

Of course it is possible to have total destruction of this data-sync byte, but that is a very slight chance. A retry counter would judge the strip unusable if the error occurs more than a few times in succession.

2.4.9 7 - Tilt > 2.0 scans

This error occurs while the reader is in the header area of the strip if, after the coarse tilt adjustments are completed, a fine tilt adjustment is calculated and is greater than the threshold allowed. Vertical sync has already been established (the barometer of a successful read), so the chance of reading the strip on a second try, even without realignment is very good. An appropriate error message would be:

"STRIP ALIGNMENT - ADJUST AND RETRY"

Total destruction of the timing marks on the left and/or right of the strip could also cause this error, in which case the strip is rendered unusable.

2.4.10 8 - Tilt could not be measured

Upon finding the start of a strip, the reader attempts to measure how accurately the lens system is aligned to the strip. This is called coarse tilt. This error occurs when the reader is unable to calculate, in very coarse units, the angle for driving the tilt motor to align the optical system to the strip.

In most cases, this error is produced by an improperly positioned strip and can be corrected by realignment. Very poor printing or strip destruction can also produce this error, but with no chance for a successful read. The following error message is required:

"STRIP ALIGNMENT - ADJUST AND RETRY"

2.4.11 9 - Watchdog timer runout

This error occurs when the reader's internal firmware detects a time-out problem that probably indicates fault with the reader's electronics or mechanical movements.

The user should be informed of the possible mechanical problem by issuing the following error message:

"READER FAILURE - CHECK POWER AND CABLES"

This sort of message should cause the user to pick up the reader while attempting to find out what's wrong and possibly jar the reader and fix what might be faulty. It's a long shot, but at this point, it's the only hope. If this error can't be corrected, then the reader requires servicing.

2.5 Host Error Codes

2.5.1 Overview

This section serves as a guide for detecting errors not necessarily caused by the reader or strip, and providing corrective actions. The list is partial, at best, since each operating system has its own differences. We leave the definitive list up to the systems programmer designing a Softstrip System communications program.

2.5.2 Strip filename too long

This error only occurs if the strip mastering process was in error or a data strip for a different operating system is being read. For instance, Applesoft allows file names to have many more characters than the 8.3 fixed format of MS-DOS. The host program should attempt to truncate the filename or ask the user for one that is appropriate for the operating system in use.

2.5.3 No filename on strip

This is an unrecoverable error. A bad strip master would be the cause.

2.5.4 Bad transmission

This error occurs when the transmitted strip length does not match the actual number of characters received. If the reader firmware signals a successful read operation, the problem may be in communications.

The host should verify that the byte count from the strip matches the number of characters actually transmitted (not including the two length bytes).

2.5.5 Checksum error

If the host computer cannot calculate a checksum identical to the one transmitted, then a bad transmission took place. Inform the user that a reread is required by issuing the message:

UNABLE TO READ - RETRY

2.5.6 Not a Softstrip data strip

Only standard Softstrip data strips are supported by the Cauzin communications programs. Special "keyed" strips will be used under applications control in customized software. Refer to the sections on Strip Type and Expansion Bytes under Field Descriptions for more information.

2.5.7 Not an "Apple DOS 3.3" data strip

This message is tailored to fit the operating system under which the communications program is running. It indicates that the data strip was not intended for use with the present operating system.

If, for example, you try to read an IBM strip into Apple DOS 3.3, this type of message would appear. The communications program could then give the user an option of saving the data in a file that is compatible with the host operating system.

2.5.8 Wrong strip ID

This message would appear if the host determines that the wrong strip was inserted and read in a multi-strip sequence. Data strips are numbered and grouped together to avoid this problem. The user must place the appropriate strip under the reader or abort this multi-read operation.

File cleanup of any partially written files should take place if the operation is aborted.

2.5.9 Strip out of sequence

The user placed a strip out of sequence during a multi-strip read. The host must insist on the proper order of insertion. If the user is unable to comply, then the read operation can be aborted and any partial files cleaned up from the disk.

2.5.10 Disk write protected

The host needs to detect a write protected data disk and advise the user of the problem and nature of action to take.

2.5.11 Disk I/O error

This error is caused by no disk in the disk drive, a bad disk, or an unformatted disk. The host needs to inform the user of the problem and allow insertion of a properly formatted data disk. Optionally, the user could be given the capability to initialize/format a data disk.

2.5.12 Highlighted files exist

This message lets the user know that these filenames already exist on the data disk. The user should be given an option of overwriting the identical filenames, or inserting a different data disk.

2.5.13 Not enough space on data disk

If not enough room can be found to place all the files from a strip onto a data disk, then this message is issued. Because many files call other files when run, it is not appropriate to allow the user to split files among more than one data disk.

3 Field Specifications

3.1 Overview

Softstrip data strips consist of one or multiple strips containing all the information necessary to reconstruct the encoded file onto a disk. Densities of the strips can be varied in the vertical and horizontal directions to match the printing industry's wide choice of papers and inks. Newspapers require low densities, while full-color magazines can use higher densities. The internal firmware of the reader automatically recognizes the density and adjusts to it. The host computer simply communicates and interprets file data.

Strips are coded with ID fields to help verify that the proper strip is being read in a multiple data-strip sequence. These strip ID's must be identical for the communications program to accept and concatenate file information. Strips also receive sequence numbers so that file information won't be read out of order. It is the responsibility of the host communications program to verify both the strip ID and the sequence number as strips are read.

3.2 Standard Field Layout

Softstrip Field Layout

Field	Description	Length	Attributes
1.	Horiz Sync		
2.	Vertical Sync		
3.	Data Sync	1	$00
4.	Expansion Bytes	2	$00 See $00 Extended Description
5.	Length lsb msb	2	Two-byte hex length of total bytes following this field
6.	Checksum	1	Binary ADD with carry
7.	Strip I.D.	6	A mandatory 6-byte ID field preferably ASCII
8.	Sequence No.	1	Binary
9.	Strip Type	1	Binary
10.	Software Expansion	2	$0000 Binary (reserved)
11.	Op. Sys. Type	1	Binary
12.	Number Files	1	Binary
13.	Cauzin Type	1	Binary
14.	O.S. FileType	1	Binary
15.	File Length	3	Binary lsb, nsb, msb order
16.	Filename	Var.	ASCII, variable length name
17.	Terminator	1	$00/FF filename terminator
18.	Block Expand	Var.	$00 defaults 1 byte no info.
19.	Data	Var.	File data information in consecutive blocks based on previous file lengths.
20.	CRC	2	optional, at strip end

Fields 1 to 10 must be on every strip. Fields 11 to 18 occur only on the first strip in a sequence. Fields 13 to 18 are repeated as necessary for each file on the logical strip (which may be any number of physical strips).

3.3 Reader Output

The sequential order of bytes transmitted from the reader to the host, after the initial request to read is established, is everything from field number 5 to the end of the strip. Fields 1 through 4 are used by the reader for initialization and alignment and are not transferred to the host.

Remember that the file description fields will repeat as needed for each file on the strip.

There is no provision at this time to allow the strip's file directory to cross strip boundary.

3.4 Field Descriptions

3.4.1 Horizontal synchronization

The horizontal sync field starts at the beginning of every data strip. It provides three basic functions:

1. Provides "coarse" alignment for the reader as it begins the scanning process. If alignment can't be established, then the reader informs the host with an appropriate error message.

2. Serves as a "densitometer" to measure the difference between black and white. The reader's electronics are automatically adjusted to compensate for dark or light inks used in printing a data strip.

3. Establishes the "data density" of this strip. Strips are printed in various byte densities, and the reader automatically adjusts itself for each strip. This allows printing "low" byte-density strips on marginal paper stock such as newsprint, and "high" byte-density strips on better calendared paper found in many magazines.

Note: during the horizontal synchronization phase only an error causes any communication to occur with the host computer. All other functions take place within the reader's microprocessor.

3.4.2 Vertical synchronization

The vertical synchronization field immediately follows the horizontal field on all Softstrip data strips. It provides the following functions to the reader's internal microprocessor.

1. Establishes the vertical scanning density. Each data line on the strip is made long enough in the vertical dimension to allow the reader to scan it several times, and the reader needs to know how many scans it will make on the same data line before a new data line occurs. This allows us to customize strip density for printing purposes, while ensuring that the reader will not send duplicate data, no matter how many times the same data line is scanned. If the reader is unable to find the vertical density, an error message is sent to the host computer.

2. Provides a "fine" tilt measurement for the reader to use to align the optical system so that the scan line is parallel to the data line.

3.4.3 Data synchronization

This field provides the reader with a method of determining that vertical synchronization is over and that strip information to be deciphered is about to start. This byte contains a zero ($00), which is not a valid byte within the vertical synchronization.

If the reader is unable to establish "data sync" it issues an error message to the host computer.

3.4.4 Expansion bytes

These two bytes were originally intended for future expansion of the Softstrip System. As the saying goes, "the future is upon us", and the bytes have been defined as follows:

The first byte is used to determine if a given strip is usable by a particular reader. It differs from the Strip Type byte in that the decision is made in hardware by the reader and, if this byte does not match the one coded in to the reader's firmware, the read will terminate immediately. The current production version of the reader requires this byte to be $00, otherwise an error is sent to the host.

To make use of this feature requires modifications to the reader's CPU board and processor firmware. These modifications are available from Cauzin at a premium.

The value of the second byte determines how many bytes to "skip" or be absorbed internally within the reader for specialized computation. For example, if the second byte is five ($05) then the next five bytes on the strip will not be sent to the host. It will be read by the reader and discarded. This feature is reserved by Cauzin. This byte must be made $00 in all data strips to ensure future compatibility.

3.4.5 Length

This two-byte field contains the total number of bytes that the reader will be sending to the host, not including the length bytes themselves. The field effectively contains the byte count of fields 6 thru 20.

The host computer's software should compare the length against an actual byte count of the transmitted data to make sure that no data has been lost. It is the duty of the host computer to insure that transmission is OK, and if a discrepancy occurs to ask for another transmission (read).

3.4.6 Checksum

The checksum is a one-byte ADD with carry of every byte that follows on the strip. After the calculation is completed, a two's complement is performed. This one-byte number is the checksum. Even an optional CRC check at the strip's end would be included in this calculation.

The host communications program is responsible for calculating a checksum against every byte received (not including the length bytes or the checksum itself) and then comparing it to the checksum byte. It is the function of the host to flag a bad transmission due to checksum mismatch and request a reread of the strip.

3.4.7 Strip ID

The six-byte strip ID identifies the data strip's name. Any characters suffice and this is a mandatory field. Sequential strips must be checked to see that all strips in the group have the same strip ID.

Preferably, the strip ID is entered in printable ASCII when the strip is mastered at the Cauzin facility in Waterbury, or by any of the Stripper (tm) strip generating products available from Cauzin, but any characters are OK. The major reason for having a strip ID is to verify that the correct data strip is being scanned in a sequence of strips.

For example: A series of five strips make up one program and the strip ID is "NIBBLE." If the host requests strip #2 in the sequence and it reads an ID of "BYTE ", then it displays a message that the wrong strip was inserted.

It is the responsibility of the host communications program to keep track of strip ID's and recognize an error in insertion. The reader's firmware does not check this condition.

3.4.8 Sequence number

The sequence number is a 7-bit binary number indicating this strip's position in a series of strips.

It is the responsibility of the host communications program to identify an out of sequence strip and issue an appropriate message to the host operator to insert the proper strip.

3.4.9 Strip type

This one-byte field identifies the strip format. Strips can be used for special purposes (such as data, program protection, or encryption), and can be used with nonstandard and non-Cauzin communications programs. This field provides a mechanism for creating these nonstandard strips and protect them from being read in to the wrong environment. This differs from the first expansion byte (refer to the section on Expansion Byte Field Description) in that a non-valid strip type will not terminate a read.

Value	Function
$00	Standard Softstrip data strip For use with the Cauzin strip specification and compatible communications programs.
$01	Special key strip This specification is not yet defined. The intent is to protect programs until the "key" strip is read.
$02-FF	Other formats These are not yet defined.

3.4.10 Software expansion

The first of these two bytes is defined as a flag-byte and the MSbit(7) denotes whether a CRC check is being done on the strip. The other 7 flag bits are undefined.

If a CRC is present, it is stored as the last two bytes on a strip, is included as part of the strip length, and is included in the checksum. The CRC algorithm has not yet been determined.

The second of these two bytes is reserved for future expansion.

3.4.11 Operating system

This one-byte field describes the host operating system that is required to properly utilize this data strip. As operating systems are supported, this list will grow.

Cauzin communications programs cross-support a Cauzin generic text-file and convert it to the appropriate operating system. Strip type would be $00 denoting Cauzin generic and the file type would indicate text file.

$00 - Cauzin generic strip format
$01 - COLOS - (Cauzin's Own Little Operating System)
(by Mad River Research, a Cauzin subsidiary)
$10 - Apple DOS 3.3
$11 - Apple ProDOS
$12 - Apple CPM 2.0
$14 - PC/MS-DOS (2.1)
$15 - Macintosh - MacBinary
$20 - Reserved (will be accepted as PC/MS-DOS)

All other operating system types are currently undefined. It is the duty of the host communications package to verify the operating system requirement.

3.4.12 Number of files

This byte contains the number of files in this strip's directory. For a normal Softstrip data strip, the minimum value is one ($01) to indicate one file, and the maximum value is 255 ($FF). As noted before, there is no provision currently to allow the strip file directory to cross strip boundaries.

3.4.13 Cauzin file type

This one-byte "generic" file type provides the communications program with information regarding cross operating-system transfers. This field heads each file directory entry on this strip and is ignored if the strip's operating-system type matches that of the host computer.

The field provides the information needed in future applications where, for instance, an IBM text file is read on an Apple, or an IBM BASIC program that's not in tokenized form is read into an Apple II.

This table provides some preliminary definitions. Codes not listed are still undefined.

$00 - Other/unknown/don't care
Exception: If the operating-system file type is "Cauzin" (generic) then this field identifies a text file.
$01 - Text file
$02 - Binary, executable file; object code
$04 - BASIC, tokenized code
$10 - Compressed data using Cauzin's proprietary technique (in development)

3.4.14 Operating system file type

This one-byte field provides the file type specification for the operating system field. The codes are generally taken directly from the reference manuals associated with a particular machine's Disk Operating System. For instance, the Apple's codes are the same as those needed to create the appropriate files using Apple's file manager.

Since the MS-DOS operating system handles files somewhat differently from the Apple and all files are handled in a similar manner, we have separated those files that are "executable" from those that are not.

Apple DOS 3.3 Files

$00 - Text file
$01 - Integer BASIC file
$02 - Applesoft BASIC file
$04 - Binary file
$10 - Relocatable object module file
$20 - A type file (not supported)
$40 - B type file (not supported)

Apple ProDOS Files

$04 - ASCII text file
$06 - General binary file
$FA - Integer BASIC file
$FC - Applesoft BASIC file
$FE - Relocatable object module file
$FF - System file

IBM PC-DOS/MS-DOS

$00 - Executable DOS file (.EXE, .COM, .BAT)
$01 - Any other DOS file (i.e., BASIC, text file, Lotus template, etc.)

Macintosh

$00 - MacBinary: Macintosh binary-transfer format. MacBinary data is a 128-byte header to the Cauzin strip file data. Prior fields regarding a Mac file are then ignored as required to support the MacBinary format.

$01 - Data fork text file document, non MacBinary

3.4.15 File length

File length is a three-byte field that contains the file's length in LSB, NSB, MSB order. This allows for a file with more than 16 million bytes on a strip sequence (which is unlikely). However, it does allow for more than 65,000 bytes, which may be practical on a multi-strip sequence.

A file's length is defined as all file-data bytes not part of the directory. For the Macintosh, this length includes the MacBinary 128-byte header, which is then interpreted by the appropriate communications program running on the host.

Thus, a file with 35,870 bytes would have its length coded as: $1E 8C 00.

3.4.16 Filename

The variable-length filename field is used in recreating the filename on disk. Preferably ASCII, any filename valid for the appropriate operating system will do.

The filename is terminated by the following field, that contains either a hexadecimal $00 or $FF not part of the filename. The host system must verify the correctness and length of a filename, issuing any warnings or error messages to the strip user.

3.4.17 Filename terminator

This one-byte field is required for terminating the prior variable length field. It not only identifies the end of a filename, but can set an executable flag option. If set, the host then asks the user if they wish to run one of the programs just read.

$00 - terminate filename
$FF - terminate filename, signal executable option

This flag in no way affects the ability of any file on a multiple strip to be executed by the user. This option only puts the burden on the host communications program to display a selected file for possible "auto"-execution by the user without leaving the communications program.

3.4.18 Miscellaneous info

This variable-length field block is reserved for future expansion of file directory information. The first byte of this field contains a count of how many bytes following the first one make up the variable field.

Currently, Cauzin is investigating the Apple ProDOS directory format and may use this area to include additional directory information required by ProDOS.

As of this writing, no format has been defined and all strips are produced with $00 to indicate no field content.

3.4.19 File data

This field is composed of all file data following the completion of the file directory fields. Since all file directory fields are grouped together at the beginning of the first strip, the length of this field can be calculated (crossing strips if necessary) by adding field #15 of each file directory.

Each file on a strip carves out a portion of this field utilizing its own byte count. File information is joined together on strips with no separation or indicator other than using file length.

If the CRC checking flag is set, then compensation must be made in the host for those two bytes that are found at that strip's end. They are not included as part of the file information.

Note: in a multiple strip sequence, some strips can be tagged for CRC, while others may not be. Also, the host could ignore the CRC by not verifying it, although the two bytes must be accounted for.

3.4.20 Optional CRC

A two-byte (16-bit) Cyclic Redundancy Check provides a 99% chance of catching transmission errors. If the flag option is set (refer to the Software Expansion Field Description), this field will occur as the last two bytes on a strip.

The host has the option of calculating a CRC for comparison or ignoring it. This feature is not currently implemented in the Cauzin communications programs.

4 Other Start-Up Commands

4.1 Scan-Only Mode

4.1.1 Overview

The scan-only mode was set up to allow for future applications utilizing the reader as a scanning device.

A digitized pre-amp output signal (SLICE) containing the digital waveform of the image being scanned can be sampled by a host computer and an application program can then manipulate this data to perform mark-sense, character recognition, or any other application one can think of. A special cable is needed, since standard Cauzin cables don't connect this line. The signal is available at the reader connector as defined below.

4.1.2 Sequence of events

To activate the scan-only mode, the host transmits an "H". The reader acknowledges the command as described above. After that, the host must send three $0E's as attributes to the "H" command. These attributes are for future use and any other sequence could damage the reader and cause unpredictable results.

HOST	READER
"H" --->	
	<--- CTF (Break)
	<--- "H"
$0E --->	
$0E --->	
$0E --->	

Note that the attributes are not echoed, only the command is.

The reader will then scan for 3599 scans (3599 * .0025 = 8.9975 inches) at 15 milliseconds per scan (about 54 seconds). During this time, the reader's data out line pulses low between scans and stays high during scans.

4.1.3 Connector information

PIN	NAME	DESCRIPTION
1	LOW RET	SIGNAL RETURN FOR 25MV (CASSETTE) DATA IN
2	DATA IN	SERIAL DATA INTO READER FROM HOST
3	CASSETTE/RS-232	SELECT CASSETTE(1) OR RS-232(0) MODES
4	SLICE	DIGITAL REPRESENTATION OF OPTICAL SIGNAL
5	DATA OUT	SERIAL DATA FROM READER TO HOST
6	GND	GROUND (SIGNAL GROUND FOR RS-232)

NOTES:

1. Pin 3 is pulled up (1) internally. Ground it (0) for RS-232.

2. SLICE is not connected on any cable presently available from Cauzin. Six-conductor modular cables are available from:

PACER ELECTRONICS (617) 935-8330
70 HOLTON STREET TERRI GOUR
WOBURN, MA 01801

5', 6 POS. MODULAR TO 6 POS, MODULAR: PART #05-665-0
They were $2.92 as of this writing.

4.2 ID Mode

ID mode is useful for interrogating the reader to determine the version of the operating firmware. No data strip is required and the ID command provides a clean method for verifying that the host and the reader are communicating.

Host	Reader
"I" --->	
	<--- CTF (Break)
	<--- "I"
	<--- "1" version number - "1" at this writing
	<--- "." a period always
	<--- "0" revision number - "0" at this writing

4.3 Terminal Mode

Terminal mode was installed to let the reader function under a somewhat simple ASCII terminal protocol. Since the command codes used in normal Cauzin communications can do disastrous things to "standard," commercial communications programs (i.e., modem versions), this mode turns off most commands and error handling.

Activation is accomplished by transmission of a "T" or "t" to the reader. The reader sends no acknowledgment except to start itself up and read a strip. The first 13 characters on the strip are absorbed and transmission starts with field #11, Op Sys Type.

The usefulness of this mode may be suspect, but it would be possible for a commercial product (e.g., PC-TALK) to capture an ASCII BASIC file from a strip, perform some editing, and utilize a strip-based program without using a Cauzin communications program.

4.4 Test Mode

The test function is only useful in quality assurance of the reader and requires ancillary programs. It is activated by transmitting a "C" command. The reader then sends timing and synchronization data to the host.

4.5 Bit-Test Mode

This function is used in quality assurance of the reader and requires ancillary programs. It is activated by transmitting a "B" command. Depending on the strip density and quality, either a reader can be evaluated for it's dependability, or a strip can be evaluated for its "readability."

In STRIPPER (Cauzin's strip printing program), this test mode is used to rate the quality of a strip (i.e., the ability of a given printer and ribbon to make a data strip).

The strip must be 4-5 inches long to be used in performing a valid test. Bad DiBits indicate that some error correction by the scanner may be needed. It's an overall indication and not necessarily conclusive regarding a strip's condition.

5 Cauzin Communications Programs

5.1 Current Versions

Cauzin currently supports the following brands of personal computers and operating systems:

IBM PC, AT, XT and other 100% IBM compatibles running PC-DOS V2.1 or greater.

Apple Computer Apple II, II Plus, //e, //e Enhanced, and //c running either DOS 3.3 or ProDOS.

Apple Computer Macintosh 128, 512 and PLUS.

Accessory kits containing a disk, manual, and required cables for all of the above are available from your local Cauzin dealer or directly from Cauzin.

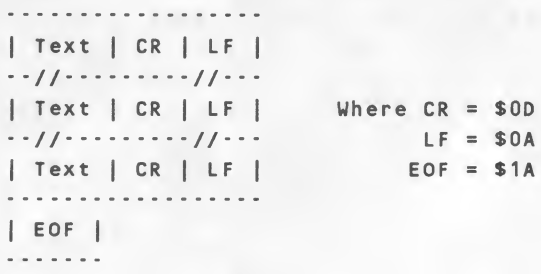
6 Cauzin Generic Text Data Strips

6.1 Overview

Each host computer stores text files in a different format. To provide inter-host text file transfers by data strip, Cauzin has defined its own proprietary format that is freely distributed in the hope of establishing a standard.

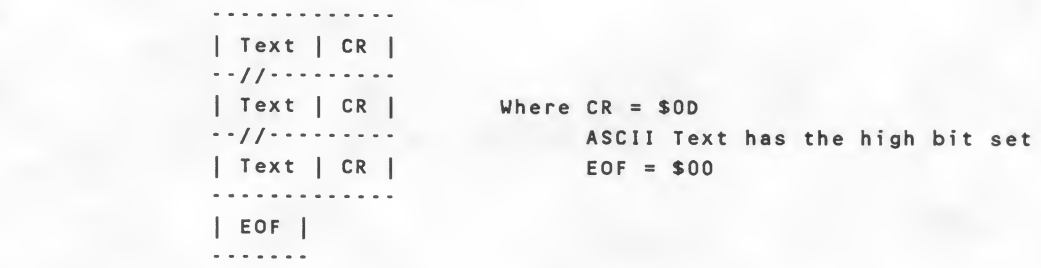
It is the duty of the host communications program to convert the Cauzin text file from data strip to the appropriate format for the host operating system.

The Cauzin text file resembles an IBM PC-DOS or MS-DOS text file and consists of the following format:



6.2 Apple DOS 3.3

An Apple DOS 3.3 text file consists of data organized in the following format:



The Cauzin generic format has three major differences from the Apple DOS 3.3 format:

1. The high bit is not set
2. There is a linefeed following each carriage return
3. The EOF character is a $1A instead of a $00

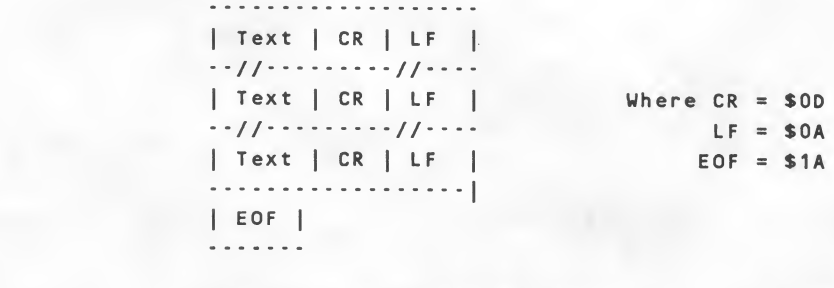
The Apple communications program needs to make these changes:

1. Set the high bit on ASCII characters
   including the carriage return ($0D --> $8D).
2. Eliminate the linefeed character ($0A)
   whenever it occurs with a carriage return.
3. Convert the EOF character from a $1A to a $00.

These changes require modification to the file length when the file is stored to disk.

6.3 IBM PC DOS 2.1

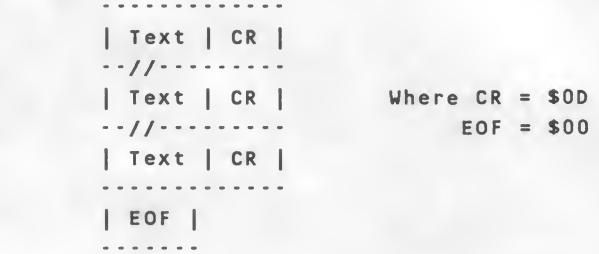
An IBM PC-DOS or MS-DOS text file consists of data organized in the following format.



Cauzin's text file is identical to the PC-DOS format, so the communications program need only be able to identify this as a legitimate Cauzin generic data strip.

6.4 Macintosh

An Apple Macintosh text file consists of data organized in the following format:



The Cauzin generic format has two major differences from the Macintosh format:

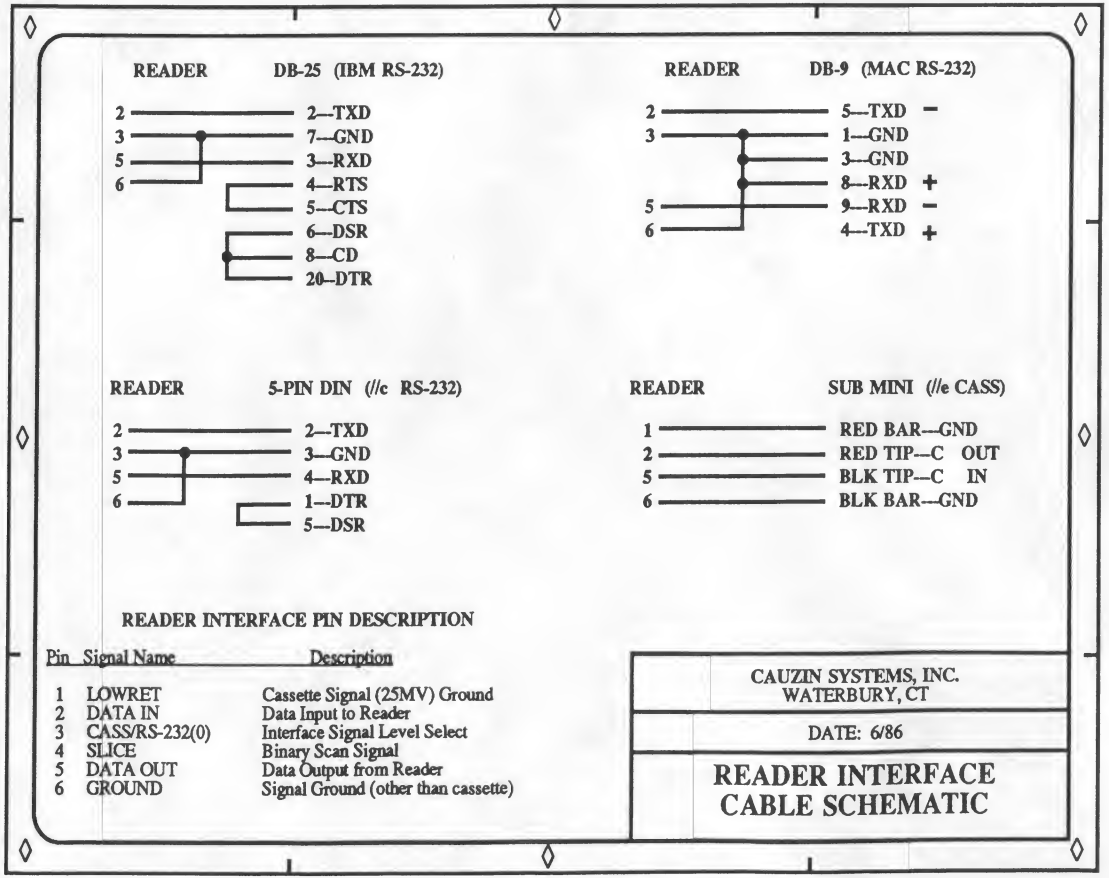
1. There is a linefeed following each carriage return

2. The EOF mark is a $1A

The Macintosh communications program needs to make these changes:

1. Eliminate the linefeed character ($0A)
   whenever it occurs with a carriage return.

2. Make the EOF a $00


READER DB-25 (IBM RS-232)
2 2—TXD
3 7—GND
5 3—RXD
6 4—RTS
5—CTS
6—DSR
8—CD
20—DTR
READER DB-9 (MAC RS-232)
2 5—TXD −
3 1—GND
3—GND
8—RXD +
5 9—RXD −
6 4—TXD +
READER 5-PIN DIN (//c RS-232)
2 2—TXD
3 3—GND
5 4—RXD
6 1—DTR
5—DSR
READER SUB MINI (//e CASS)
1 RED BAR—GND
2 RED TIP—C OUT
5 BLK TIP—C IN
6 BLK BAR—GND
READER INTERFACE PIN DESCRIPTION
Pin Signal Name Description
1 LOWRET Cassette Signal (25MV) Ground
2 DATA IN Data Input to Reader
3 CASS/RS-232(0) Interface Signal Level Select
4 SLICE Binary Scan Signal
5 DATA OUT Data Output from Reader
6 GROUND Signal Ground (other than cassette)
CAUZIN SYSTEMS, INC.
WATERBURY, CT
DATE: 6/86
READER INTERFACE
CABLE SCHEMATIC